

AMENDMENTS TO THE CLAIMS

1. (Previously Presented) A method for exchanging data between a source location and a destination location comprising the steps of:

generating a data file formatted in a markup language in accordance with a predetermined schema;

generating a first software envelope containing the data file;

selecting a plugin from a plurality of plugins based on the predetermined schema, said plugin configured to create an object from the markup language of the data file;

attaching the plugin to the software envelope;

transmitting the software envelope to the destination location; and

creating the object from the markup language of the data file with the plugin,

wherein the step of creating the object from the markup language of the data file with the plugin comprises

parsing the data file into a plurality of data fields according to the predetermined schema of the plugin,

determining a first data field in the plurality as an optional data field based on the predetermined schema, and

determining a second data field in the plurality as a required data field based on the predetermined schema.

Claims 2-4 (Canceled)

5. (Original) The method of claim 1, wherein the markup language comprises extensible markup language (XML).

6. (Original) The method of claim 1, wherein the markup language comprises standard generalized markup language (SGML).

7. (Original) The method of claim 1, wherein the step of transmitting comprises transmitting the software envelope via electronic mail.

8. (Original) The method of claim 1, wherein the step of transmitting comprises transmitting the software envelope via HTTP.

Claims 9-24 (Canceled)

25. (Previously Presented) The method of claim 1, wherein the software envelope contains the plugin.

Claim 26 (Canceled)

27. (Currently Amended) A method for exchanging data between a source location and a destination location comprising the steps of:

generating a data file formatted in a markup language in accordance with a predetermined schema;

generating a first software envelope formatted in a markup language and containing the data file, the first software envelope including fields for forward and return routing information;

selecting a plugin based on the predetermined schema, said plugin configured to create an object from the markup language of the data file;

determining an operating system associated with the destination location before selecting ~~generating~~ the plugin, wherein the plugin is selected based on said operating system;

attaching the plugin to the first software envelope; and

transmitting the first software envelope to the destination location,

wherein the first software envelope is formatted to allow an application in the destination location to automatically generate a reply software envelope for routing information to the source location by swapping the forward and return routing information in the respective fields of the first software envelope, and

wherein the plugin includes executable code to provide to the application the functionality of creating ~~creates~~ the object from the markup language of the data file.

28. (Previously Presented) The method of claim 27, wherein the markup language comprises extensible markup language (XML).

29. (Previously Presented) The method of claim 27, wherein the markup language comprises standard generalized markup language (SGML).

30. (Currently Amended) The method of claim 27, wherein the step of transmitting comprises transmitting the first software envelope via electronic mail.

31. (Currently Amended) The method of claim 27, wherein the step of transmitting comprises transmitting the first software envelope via HTTP.

32. (Currently Amended) The method of claim 27, wherein the first software envelope contains the plugin.

33. (Previously Presented) One or more computer readable media storing computer-executable instructions which, when executed on a computer system, cause the computer to perform a method comprising steps of:

generating a data file formatted in a markup language in accordance with a predetermined schema;

generating a first software envelope containing the data file;

selecting a plugin from a plurality of plugins based on the predetermined schema, said plugin configured to create an object from the markup language of the data file;

attaching the plugin to the software envelope;

transmitting the software envelope to the destination location; and

creating the object from the markup language of the data file with the plugin,

wherein the step of creating the object from the markup language of the data file with the plugin comprises

parsing the data file into a plurality of data fields according to the predetermined schema of the plugin,

determining a first data field in the plurality as an optional data field based on the predetermined schema, and

determining a second data field in the plurality as a required data field based on the predetermined schema.

34. (Previously Presented) The computer readable media according to claim 33, wherein the markup language comprises extensible markup language (XML).

35. (Previously Presented) The computer readable media according to claim 33, wherein the markup language comprises standard generalized markup language (SGML).

36. (Previously Presented) The computer readable media according to claim 33, wherein the software envelope contains the plugin.

37. (Currently Amended) One or more computer readable media storing computer-executable instructions which, when executed on a computer system, cause the computer system to perform a method comprising steps of:

generating a data file formatted in a markup language in accordance with a predetermined schema;

generating a first software envelope containing the data file, the first software envelope including fields for forward and return routing information;

selecting a plugin based on the predetermined schema, said plugin configured to create an object from the markup language of the data file;

determining an operating system associated with the destination location before selecting generating the plugin, wherein the plugin is selected based on said operating system;

providing an identifier of the selected attaching the plugin in to the first software
envelope; and

transmitting the first software envelope to the destination location,
wherein the first software envelope is formatted to allow an application in the destination
location to automatically generate a reply software envelope for routing information to the
source location by swapping the forward and return routing information in the respective fields
of the first software envelope, and

wherein the plugin identifier identifies executable code in the destination location to be
plugged into the application to provide the functionality of creating ~~creates~~ the object from the
markup language of the data file.